

**App ndix**

```

entity top is
  port (
    clk: in std_logic;
    din: in std_logic_vector(7 downto 0);
    dout: out std_logic_vector(7 downto 0)
  );
end top;

architecture structural of top is
  component clock_driver
    port (
      clk: in std_logic;
      ce1: in std_logic;
      phase: out std_logic;
      ce2: out std_logic
    );
  end component;
  component middle
    port (
      clk: in std_logic;
      ce1: in std_logic;
      ce2: in std_logic;
      phase: in std_logic;
      din: in std_logic_vector(7 downto 0);
      dout: out std_logic_vector(7 downto 0)
    );
  end component;

  signal ce1_x_0: std_logic;
  signal ce2_x_0: std_logic;
  signal phase_x_0: std_logic;

begin
  clock_driver_x_0: clock_driver
    port map (
      phase => phase_x_0,
      clk => clk,
      ce1 => ce1_x_0,
      ce2 => ce2_x_0
    );
  middle_x_1: middle
    port map (
      phas => phase_x_0,
      clk => clk,
      ce1 => ce1_x_0,

```

```
        ce2 => ce2_x_0,
        din => din,
        dout => dout
    );
end structural;

entity middle is
    port (
        clk: in std_logic;
        ce1: in std_logic;
        ce2: in std_logic;
        phase: in std_logic;
        din: in std_logic_vector(7 downto 0);
        dout: out std_logic_vector(7 downto 0)
    );
end middle;

architecture structural of middle is
    component bottom
        port (
            clk: in std_logic;
            ce1: in std_logic;
            ce2: in std_logic;
            phase: in std_logic;
            din: in std_logic_vector(7 downto 0);
            dout: out std_logic_vector(7 downto 0)
        );
    end component;
begin
    bottom_x_0: bottom
        port map (
            phase => phase,
            clk => clk,
            ce1 => ce1,
            ce2 => ce2,
            din => din,
            dout => dout
        );
end structural;

entity bottom is
    port (
        clk: in std_logic;
        ce1: in std_logic;
        ce2: in std_logic;
        phase: in std_logic;
        din: in std_logic_vector(7 downto 0);
```

```
        dout: out std_logic_vector(7 downto 0)
    );
end bottom;

architecture structural of bottom is
    component xldsamp
        port (
            clk: in std_logic;
            src_ce: in std_logic;
            dest_ce: in std_logic;
            phase: in std_logic;
            d: in std_logic_vector(7 downto 0);
            q: out std_logic_vector(7 downto 0)
        );
    end component;
begin
    down_sample: xldsamp
    port map (
        clk => clk,
        src_ce => ce1,
        dest_ce => ce2,
        phase => phase,
        d => din,
        q => dout
    );
end structural;
```